MASSACHUSETTS INSTITUTE OF TECHNOLOGY

ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 533

June, 1979
Revised June, 1980

A Recursive Lagrangian Formulation of Manipulator Dynamics

John M. Hollerbach

ABSTRACT. An efficient Lagrangian formulation of manipulator dynamics has been developed. The efficiency derives from recurrence relations for the velocities, accelerations, and generalized forces. The number of additions and multiplications varies linearly with the number of joints, as opposed to past Lagrangian dynamics formulations with an n^4 dependence. With this formulation it should be possible in principle to compute the Lagrangian dynamics in real time. The computational complexities of this and other dynamics formulations including recent Newton-Euler formulations and tabular formulations are compared. It is concluded that recursive formulations based either on the Lagrangian or Newton-Euler dynamics offer the best method of dynamics calculation.

Acknowledgements. This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Office of Naval Research under Office of Naval Research contract N00014-77C-0389.

Introduction

The inverse problem for manipulator dynamics, namely the problem of computing the joint torques required to produce given joint positions, velocities, and accelerations, has until recently been a computational bottleneck in the control of manipulators. Past formulations of manipulator dynamics have been based on a relatively inefficient derivation from the Lagrange equations [1,2]. The solutions to these equations required on the order of seconds on a computer for each trajectory point, and it was perceived that therefore torques for trajectories could not be computed in real time.

Two different types of schemes were proposed to render the Lagrangian dynamics computationally feasible: (1) simplifying the dynamics by ignoring some terms and correcting errors with feedback, and (2) tessellating the state space along various dimensions and tabulating part of the solution. The most common method of simplifying the dynamics has been to ignore the Coriolis and centrifugal forces, which are by far the greatest computational burden in the dynamics calculation [3,4]. However, ignoring Coriolis and centrifugal forces works well only at slower speeds of movement; at fast speeds of movement the Coriolis and centrifugal forces are a major component of the dynamics [5]. The errors in the computed torque resulting from ignoring Coriolis and centrifugal forces cannot be corrected with feedback because of excessive requirements on the corrective torques [6].

Tabular solutions to the dynamics seek to achieve economies in computation time at the expense of large memories. Raibert [7] has discussed this space-time tradeoff and has examined the different dimensions along which the tabularization can be set up. Albus [8,9] represents the space extreme by including in his table all dimensions $(q, \dot{q}, \ddot{q})$, representing n dimensional position, velocity, and acceleration vectors. Since the generalized forces have the form $F_i = f(q, \dot{q}, \ddot{q})$, these forces are obtained at no computational cost by indexing the table with the desired trajectory values $(q, \dot{q}, \ddot{q})$. Raibert [10] eliminated the acceleration dimension from the

table by expressing the dynamics in the form $F_i = J(q)\ddot{q} + K(q, \dot{q})$ and tabulating the position and velocity dependent terms $J(q)$ and $K(q, \dot{q})$. Because of the large memory costs involved in the above two methods, Horn and Raibert [6] proposed a configuration space control that tabulates only the position dependent terms. Their formulation will be presented later in conjunction with a discussion of its computational complexity. The difficulties with tabular methods include (1) enormous table sizes necessitated by creating a fine enough tessellation along the various dimensions, (2) filling the entries in the table, (3) interpolating between stored values, and (4) the tables become useless if the load changes suddenly such as when an object is picked up.

This paper presents a reformulation of the Lagrangian dynamics for a manipulator that greatly reduces the computational burden and that should allow real time computation of the inverse dynamics. Specifically, there are three parts to the reformulation:

1. backward recursion of the velocities and accelerations working from the base of the manipulator to the end link,
2. forward recursion of the generalized forces working from the end link to the base of the manipulator, and
3. use of 3x3 rotation matrices instead of 4x4 rotation-translation matrices and homogeneous coordinates.

The end result of applying these methods is that the number of additions and multiplications varies linearly with the number of joints. This reformulation brings the Lagrangian dynamics into line with recently developed recursive Newton-Euler dynamics [5,11,12], whose computational complexity varies linearly with the number of joints and whose formulation contains parts 1 and 2 above. Part 3 was delineated in [12] for the Newton-Euler method, and a similar preference for 3x3 versus 4x4 matrices was expressed for reasons of computational efficiency.

The Uicker/Kahn Lagrangian Formulation

The standard formulation for manipulator dynamics is derived from Uicker [1], who used 4x4 matrices to set up the Lagrangian based dynamics for a somewhat more general linkage problem. This formulation was particularized to open loop kinematic chains by Kahn [2]. Borrowing Kahn's notation (Figure 1), the links of

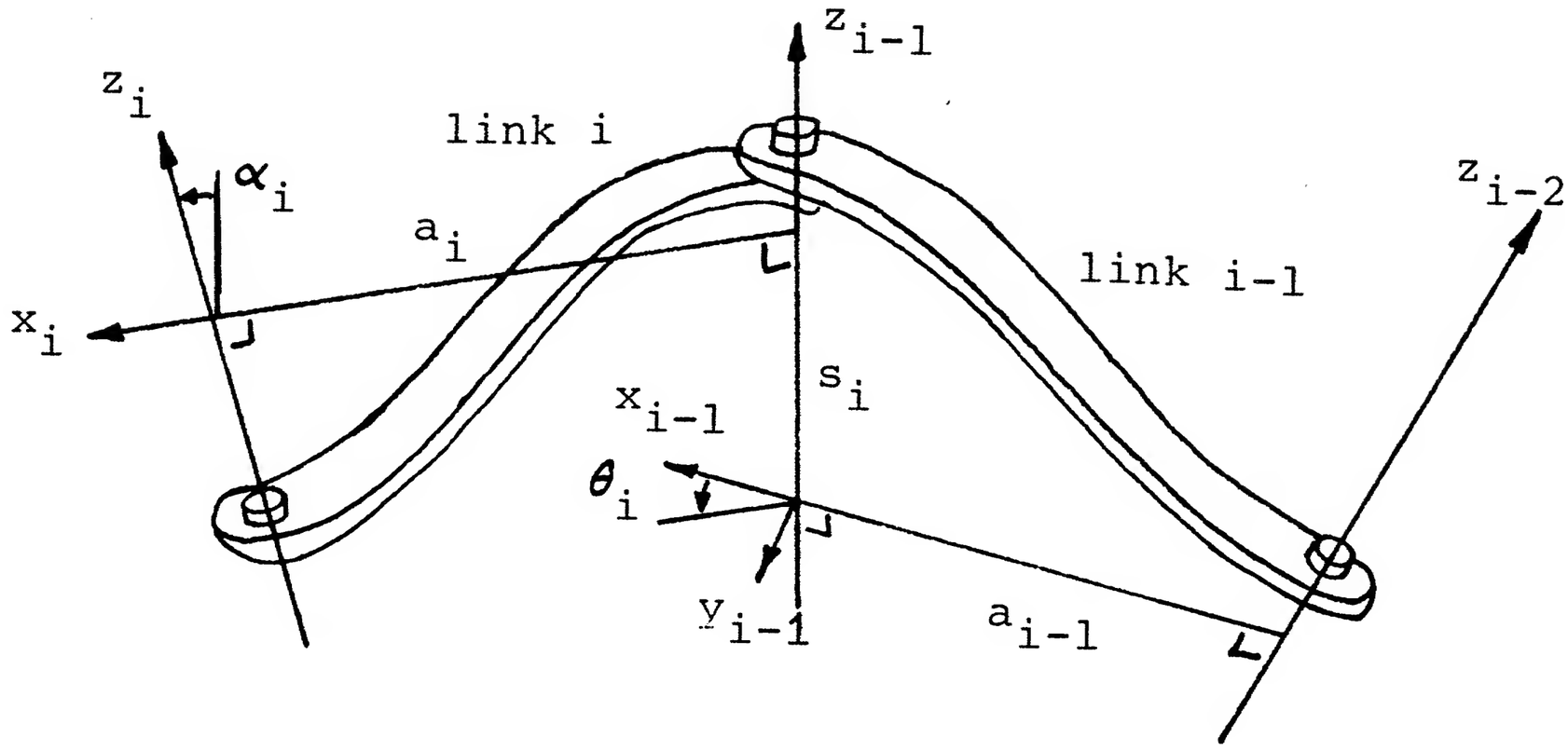

Figure 1. Standard coordinate axes definitions for connected links and relationships between neighboring coordinate axes.

a manipulator are numbered consecutively from 1 to n starting from the base to the tip. By convention the reference frame is numbered link 0. The joints are the points of articulation between links, and are numbered so that joint i connects links $i - 1$ and i. An orthogonal coordinate system is fixed in each link as follows:

z_i is directed along the axis of joint $i + 1$,

x_i lies along the common normal from z_{i-1} to z_i, and

y_i completes the right handed coordinate.

The relative position of two adjacent links is completely described by the following four parameters:

a_i is the distance between the origins of coordinate systems $i - 1$ and i measured along x_i,

s_i is the distance between x_{i-1} and x_i measured along z_{i-1},

α_i is the angle between the z_{i-1} and z_i axes measured in a righthand sense about x_i, and

θ_i is the angle between the x_{i-1} and the x_i axes measured in the righthand sense about z_{i-1}.

If the joint is rotational the joint variable will be θ_i; if translational the joint variable will be s_i. The symbol

q_i will designate the variable for joint i whether it is s_i or θ_i. The vector $Q = (q_1, q_2, \ldots, q_n)$ represents the generalized coordinates of the manipulator and completely specifies the position. Multiple rotational degrees of freedom can be expressed by joints with zero mass and zero length links.

Let ${}^{i}v_j = (1\ x\ y\ z)^T$ be a vector from coordinate system i to a point fixed in link j expressed in link i coordinates. Then adjacent coordinate systems are related by the relation:

$$ {}^{i-1}v_j = A_i\ {}^{i}v_j \tag{1} $$

where A_i is the 4x4 transformation matrix between coordinate systems $i-1$ and i:

$$ A_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a_i \cos\theta_i & \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i \\ a_i \sin\theta_i & \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i \\ s_i & 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{2} $$

We define $A_0 = I$, the identity matrix. Any two coordinate systems i and j can be related by cascading the transformations:

$$ {}^{i}v_k = {}^{i}W_j\ {}^{j}v_k \tag{3} $$

where ${}^{i}W_j = A_{i+1}A_{i+2}\ldots A_j$ for $i < j$. We define ${}^{j}W_j = I$. When referring to the base coordinate system we omit the 0 superscript, so that $v_k = {}^{0}v_k$ and $W_j = {}^{0}W_j$.

Using this notation, the generalized forces F_i for an n-link manipulator have been derived from the Lagrange equations [1,2]:

$$ F_i = \sum_{j=i}^{n} \left[\sum_{k=1}^{j} \left(tr\left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial W_j^T}{\partial q_k} \right) \right) \ddot{q}_k + \sum_{k=1}^{j} \sum_{l=1}^{j} \left(tr\left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial^2 W_j^T}{\partial q_k \partial q_l} \right) \dot{q}_k \dot{q}_l \right) - m_j g^T \frac{\partial W_j}{\partial q_i}\ {}^{j}r_j \right] \tag{4} $$

where

J_j is the inertia tensor with respect to the proximal joint of link j expressed in j's body coordinates,

m_j is the mass of link j,

g is the gravity vector, and

${}^j r_j$ is the coordinate of the center of mass of link j expressed in link j's body coordinates.

Evaluation of (4) has been considered too slow for real time computation. As an estimate of the complexity of computing the dynamics in this formulation, the total number of additions and multiplications has been calculated. In arriving at a number, the operations in (4) were carried out more or less as set forth, because any efficiencies in calculating these terms should be explicit in the formulation rather than hidden in a program. A few reasonable economies nevertheless were practiced. Simple decompositions saved some recomputation.

$$\begin{aligned}
\frac{\partial W_j}{\partial q_k} &= W_{k-1} \frac{\partial A_k}{\partial q_k} {}^k W_j && k \leq j \\
\frac{\partial^2 W_j}{\partial q_k \partial q_l} &= W_{k-1} \frac{\partial A_k}{\partial q_k} {}^k W_{l-1} \frac{\partial A_l}{\partial q_l} {}^l W_j && k < l \leq j \\
&= W_{k-1} \frac{\partial^2 A_k}{\partial q_k^2} {}^k W_j && k = l
\end{aligned} \tag{5}$$

The terms ${}^k W_j$ were computed first and stored. Then the $\partial W_j/\partial q_k$ and $\partial^2 W_j/\partial q_k \partial q_l$ terms were computed and their results also stored. The cost of computing the terms in the matrices A_j, $\partial A_j/\partial q_j$, and $\partial^2 A_j/\partial q_j^2$ was not included, since they depend on the particular manipulator configuration and because they contribute only a small linear cost to the computational complexity. Finally, no allowance has been made for any sparseness of the matrices.

Table I shows that the Uicker/Kahn formulation gives an n^4 dependence on the number of multiplications and on the number of additions. Table II shows that for $n = 6$ these numbers are 66,271 and 51,548 respectively. These large numbers of operations are too time consuming to compute in real time. Luh et al. [5] have estimated that to compute the torques for one nominal point in the trajectory requires 7.9 seconds on a

Table I. Comparison of Dynamics Formulations		
Method	**Multiplications**	**Additions**
Uicker/Kahn	$32\frac{1}{2}n^4 + 86\,5/12n^3 + 171\frac{1}{4}n^2 + 53\frac{1}{3}n - 128$	$25n^4 + 66\frac{1}{3}n^3 + 129\frac{1}{2}n^2 + 42\frac{1}{3}n - 96$
Waters	$106\frac{1}{2}n^2 + 620\frac{1}{2}n - 512$	$82n^2 + 514n - 384$
Hollerbach(4x4)	$830n - 592$	$675n - 464$
Hollerbach(3x3)	$412n - 277$	$320n - 201$
Newton-Euler	$150n - 48$	$131n - 48$
Horn, Raibert	$2n^3 + n^2$	$n^3 + n^2 + 2n$

Table II. Comparison for $n = 6$		
Method	**Multiplications**	**Additions**
Uicker/Kahn	66,271	51,548
Waters	7,051	5,652
Hollerbach(4x4)	4,388	3,586
Hollerbach(3x3)	2,195	1,719
Newton-Euler	852	738
Horn, Raibert	468	264

PDP 11/45.

Lagrangian Dynamics with Backward Recursion

Waters [13] first noticed that the generalized forces (4) could be expressed in a form that allowed one to take advantage of several backward recurrence relations (backward in the sense of the link numbering direction). We present below a simpler formulation and derivation of these recurrence relations. In reexamining the derivation of the generalized forces it becomes apparent that the generalized forces (4) can be expressed in a more compact form:

$$F_i = \sum_{j=i}^{n} \left[tr\left(\frac{\partial W_j}{\partial q_i} J_j \ddot{W}_j^T \right) - m_j g^T \frac{\partial W_j}{\partial q_i}\, {}^j r_j \right] \qquad i = 1, \ldots, n \tag{6}$$

where $\ddot{W}_j$ could be expanded to yield the usual form for the components of the reaction and the Coriolis and

centrifugal forces:

$$\ddot{W}_j = \sum_{k=1}^{j} \frac{\partial W_j}{\partial q_k} \ddot{q}_k + \sum_{k=1}^{j} \sum_{l=1}^{j} \frac{\partial^2 W_j}{\partial q_k \partial q_l} \dot{q}_k \dot{q}_l \tag{7}$$

It is however best to leave this term unexpanded. With this more compact formulation the specific recurrence relations for the velocities $\dot{W}_j$ and accelerations $\ddot{W}_j$ are easily derived by straightforward differentiation:

$$W_j = W_{j-1} A_j \tag{8}$$

$$\begin{aligned} \dot{W}_j &= \dot{W}_{j-1} A_j + W_{j-1} \dot{A}_j \\ &= \dot{W}_{j-1} A_j + W_{j-1} \frac{\partial A_j}{\partial q_j} \dot{q}_j \end{aligned} \tag{9}$$

$$\begin{aligned} \ddot{W}_j &= \ddot{W}_{j-1} A_j + \dot{W}_{j-1} \dot{A}_j + \dot{W}_{j-1} \frac{\partial A_j}{\partial q_j} \dot{q}_j + W_{j-1} \frac{d}{dt} \left(\frac{\partial A_j}{\partial q_j} \right) \dot{q}_j + W_{j-1} \frac{\partial A_j}{\partial q_j} \ddot{q}_j \\ &= \ddot{W}_{j-1} A_j + 2 \dot{W}_{j-1} \frac{\partial A_j}{\partial q_j} \dot{q}_j + W_{j-1} \frac{\partial^2 A_j}{\partial q_j^2} \dot{q}_j^2 + W_{j-1} \frac{\partial A_j}{\partial q_j} \ddot{q}_j \end{aligned} \tag{10}$$

With this formulation the number of additions and multiplications is reduced to an n^2 dependence (Tables I and II). For $n = 6$ there are 7051 multiplications and 5652 additions. The reduction from an n^4 to an n^2 dependence comes about primarily from a more efficient Coriolis and centrifugal force computation; that is to say, this formulation requires only calculation of $\partial^2 W_i / \partial q_i^2$ instead of all of the matrices $\partial^2 W_j / \partial q_k \partial q_l$.

Forward Recursive Lagrangian Dynamics

An additional level of forward recursion can be placed on the generalized force formulation (6), leading to further efficiency of the Lagrangian dynamics. We note that $\partial W_j / \partial q_i = \partial W_i / \partial q_i \; {}^i W_j$. The generalized force equation (6) can then be written:

$$F_i = \sum_{j=i}^{n} \left[tr\left(\frac{\partial W_i}{\partial q_i} {}^{i}W_j J_j \ddot{W}_j^T \right) - m_j g^T \frac{\partial W_i}{\partial q_i} {}^{i}W_j {}^{j}r_j \right]$$

$$= tr\left(\frac{\partial W_i}{\partial q_i} \sum_{j=i}^{n} {}^{i}W_j J_j \ddot{W}_j^T \right) - g^T \frac{\partial W_i}{\partial q_i} \sum_{j=i}^{n} m_j {}^{i}W_j {}^{j}r_j \tag{11}$$

This reformulation lends itself to forward recursion as follows:

$$D_i = \sum_{j=i}^{n} {}^{i}W_j J_j \ddot{W}_j^T$$

$$= {}^{i}W_i J_i \ddot{W}_i^T + \sum_{j=i+1}^{n} A_{i+1} {}^{i+1}W_j J_j \ddot{W}_j^T$$

$$= J_i \ddot{W}_i^T + A_{i+1} D_{i+1} \tag{12}$$

$$c_i = \sum_{j=i}^{n} m_j {}^{i}W_j {}^{j}r_j$$

$$= m_i {}^{i}r_i + A_{i+1} c_{i+1} \tag{13}$$

Substituting into (11), the generalized forces become simply:

$$F_i = tr\left(\frac{\partial W_i}{\partial q_i} D_i \right) - g^T \frac{\partial W_i}{\partial q_i} c_i \tag{14}$$

This recursive formulation is computed similarly to recursive formulations of the Newton-Euler dynamics [5,11,12]. First all the $\ddot{W}_i^T$ terms are computed staring from $i = 1$ to $i = n$ using the recurrence relations (6). Then the D_i and c_i terms are computed in the other direction, from $i = n$ to $i = 1$. With this formulation there are now $830n - 592$ multiplications and $675n - 464$ additions; for $n = 6$ there are 4388 multiplica-

tions and 3586 additions. This linear computation cost has essentially been obtained for free, eliminating the n^2 cost in Waters' formulation without significantly changing the coefficients of the other terms.

Recursive Lagrangian Dynamics with 3x3 Matrices

Since the previous formulation reduced the computational cost to an n dependence, all that can be done is to reduce the size of the coefficients. The greatest reduction can be obtained through a reformulation of the Lagrangian dynamics in terms of 3x3 matrices rather than 4x4 rotation-translation matrices. Although more convenient in setting up the dynamics, 4x4 matrices are inefficient because of some sparseness and because of the combination of translation with rotation (see e.g. [12]). While it is possible to have special purpose multiplication routines which take into account the zeros and ones of the last row of the 4x4 transformation matrices, it has consistently been the position of this paper that economies in computation should be explicit in the formulation rather than implicit in the programming. Because 3x3 matrix multiplications require 27 multiplications while 4x4 matrix multiplications require 64, we get a greater than 50% reduction in the coefficients of the computational cost terms.

We suppose that A_j now represents a 3x3 rotation matrix relating the orientations of coordinate system $j-1$ and j. That is to say, if ${}^jv$ is a vector expressed in terms of the orientation of coordinate system j axes,then ${}^{j-1}v = A_j\,{}^jv$. When a vector is presented without a left superscript, it is referred to the base coordinate orientation ($v = {}^0v$). In the subsequent development, capital letters represent 3x3 matrices, lower case letters represent 3x1 vectors. ${}^jW_k$ is defined as before, save that it is now the composition of 3x3 rotation matrices. Define the following vectors (see Figure 2):

p_i is a vector from the base coordinate origin to the joint i coordinate origin,

p_i^* is a vector from coordinate origin $i-1$ to coordinate origin i,

r_i is a vector from the base coordinate origin to the link i center of mass,

r_i^* is a vector from coordinate origin i to the link i center of mass, and

n_i is r_i^*/m_i.

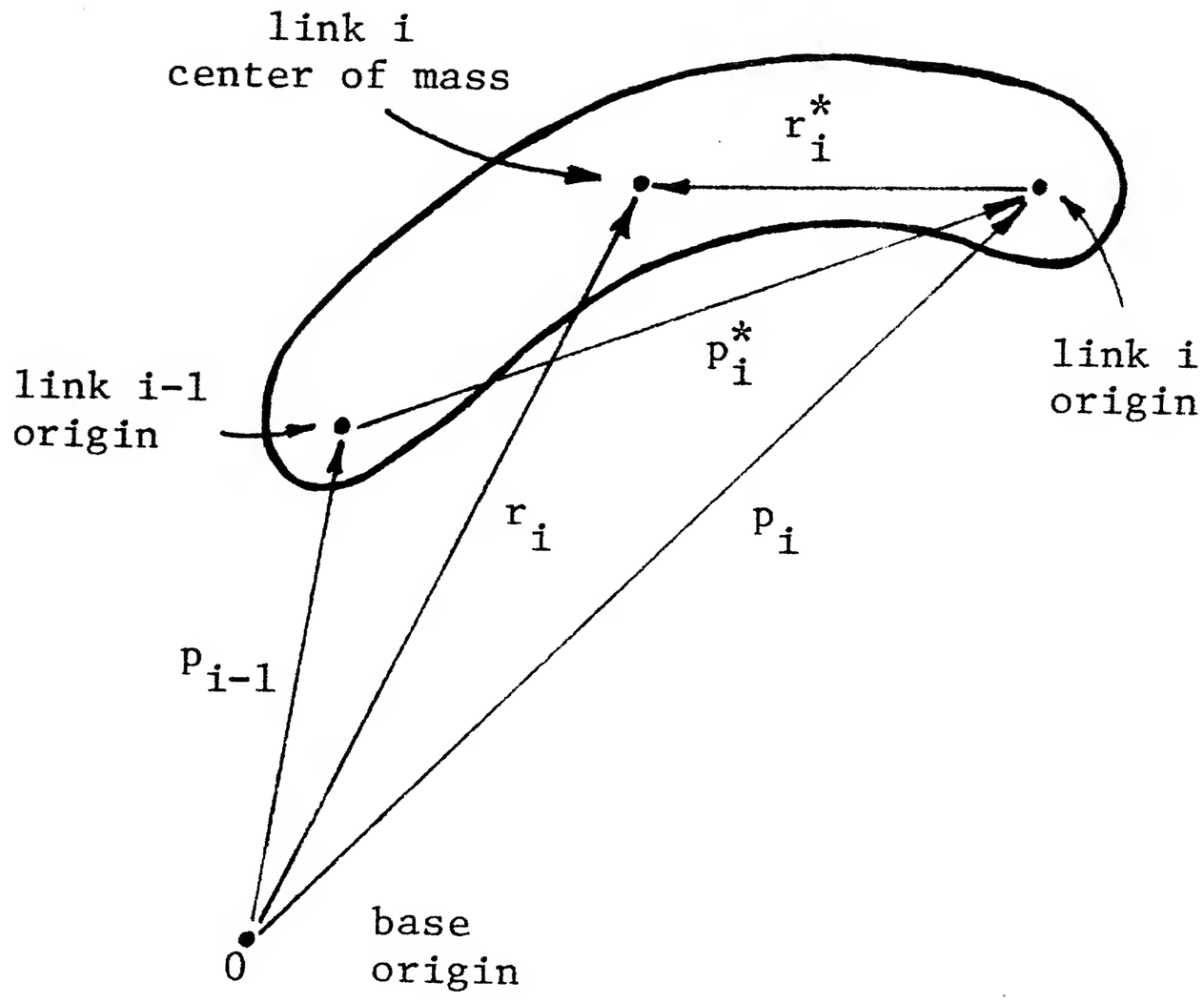

Figure 2. Vector definitions between the base origin and the link origins and center of masses.

The generalized forces for this system are derived in Appendix A:

$$F_i = \sum_{j=i}^{n} \left[tr\left(m_j \frac{\partial p_j}{\partial q_i} \ddot{p}_j^T + \frac{\partial p_j}{\partial q_i} {}^j n_j^T \ddot{W}_j^T + \frac{\partial W_j}{\partial q_i} {}^j n_j \ddot{p}_j^T + \frac{\partial W_j}{\partial q_i} J_j \ddot{W}_j^T \right) - m_j g^T \frac{\partial W_j}{\partial q_i} {}^j r_j \right] \qquad (15)$$

The $\ddot{W}_j$ term has the same recursive expression as in equation (10), though presently referring to a 3x3 rotation matrix. The $\ddot{p}_j$ term has the following recurrence relations:

$$\ddot{p}_j = \ddot{p}_{j-1} - \ddot{W}_j {}^j p_j^* \qquad (16)$$

Lastly, in Appendix B the forward recursive relation for D_i is derived. The recurrence relation for c_i is the

same as (13).

$$F_i = tr\left(\frac{\partial W_i}{\partial q_i} D_i\right) - g^T \frac{\partial W_i}{\partial q_i} c_i \tag{17}$$

where

$$\begin{aligned} D_i &= A_{i+1} D_{i+1} + {}^i p_{i+1} e_{i+1} + {}^i n_i \ddot{p}_i^T + J_i \ddot{W}_i^T \\ e_i &= e_{i+1} + m_i \ddot{p}_i^T + {}^i n_i^T \ddot{W}_i^T \end{aligned} \tag{18}$$

This formulation decreases the coefficients of the complexity polynomial of the recursive Lagrangian formulation with 4x4 matrices by better than 50% (Tables I and II). For n=6 there are now 2195 multiplications and 1719 additions.

Recursive Newton-Euler Dynamics

The Lagrange equations represent only one of several possible approaches towards deriving the dynamic equations for open loop kinematic chains. Other approaches include Newton-Euler formulations [5,11,12,14,-15,16,17,18], a Lagrangian formulation of D'Alembert's Principle [19,20], and virtual work formulations [21]. Various dynamics formulations have been contrasted and compared in [22].

Recently a number of papers have appeared on an efficient recursive Newton-Euler formulation of manipulator dynamics [5,11,12]. Hooker and Margulies [14] and Hooker [15] presented an early application of Newton-Euler equations to open-loop kinematic chains such as satellites. Stepanenko and Vukobratovic [16] developed a recursive Newton-Euler formulation in the context of human limb dynamics. This formulation was revised and elaborated in subsequent papers [17,18].

We present a revision of the Newton-Euler equations as presented in [5]. The backward recursion propagates angular velocities, angular accelerations, linear accelerations, total link forces, and total link torques from the base to the end link. For a rotational joint, the recursive equations are:

$$\omega_i = \omega_{i-1} + z_{i-1}\dot{q}_i$$

$$\dot{\omega}_i = \dot{\omega}_{i-1} + z_{i-1}\ddot{q}_i + \omega_{i-1} \times z_{i-1}\dot{q}_i$$

$$\ddot{p}_i = \dot{\omega}_i \times p_i^* + \omega_i \times (\omega_i \times p_i^*) + \ddot{p}_{i-1}$$

$$\ddot{r}_i = \omega_i \times (\omega_i \times r_i^*) + \dot{\omega}_i \times r_i^* + \ddot{p}_i$$

$$F_i = m_i \ddot{r}_i$$

$$N_i = I_i \dot{\omega}_i + \omega_i \times (I_i \omega_i)$$

where previously undefined terms are:

ω_i is the angular velocity of link i,

$\dot{\omega}_i$ is the angular acceleration of link i,

F_i is the total external force on link i,

N_i is the total external torque on link i, and

I_i is the inertia tensor of link i about its center of mass.

The forward recursion propagates the forces and moments exerted on link i by link $i - 1$ from the end link of the manipulator to the base.

$$f_i = F_i + f_{i+1}$$

$$n_i = n_{i+1} + N_i + (p_i^* + r_i^*) \times F_i + p_i^* \times f_{i+1}$$

$$\tau_i = z_{i-1} \cdot n_i$$

where

f_i is the force exerted on link i by link $i - 1$,

n_i is the moment exerted on link i by link $i - 1$, and

τ_i is the input torque at joint i.

In this formulation the implicit reference coordinate frame is the base coordinates. In the next section a more efficient reference frame is examined.

Recursive Newton-Euler Dynamics Referred to Link Coordinates

Orin et al. [12] initially proposed that the forces and moments in the Newton-Euler formulation of [16] be referred to the link's internal coordinate system. Armstrong [11] and Luh et al. [5] extended this idea by calculating the angular and linear velocities and accelerations in link coordinates as well. The end result of referencing both the dynamics and kinematics to the link cordinates is to obviate a great deal of coordinate transformation and to allow the inertia tensor to be fixed in each link coordinate system.

Although Armstrong's and Luh's formulations are equivalent, they addressed complementary problems. Armstrong sought to integrate the dynamic equations to find the accelerations, given the applied torques. Luh solved the inverse problem, namely to find the appropriate joint torques given desired joint positions, velocities, and accelerations. Another difference is that Armstrong's dynamics are referred to coordinate systems located at the joints, while Luh's dynamics are referred to coordinate systems located at the link centers of mass.

Using Luh's formulation once again, and rather than rewriting all the Newton-Euler equations showing how they are referred to internal link coordinates, we present 3 examples of this reformulation. For a more complete presentation, the reader is referred to [5].

$$
\begin{aligned}
{}^{i}\omega_i &= A_i^T({}^{i-1}\omega_{i-1} + {}^{i-1}z_{i-1}\dot{q}_i) \\
{}^{i}N_i &= {}^{i}I_i{}^{i}\dot{\omega}_i + {}^{i}\omega_i \times ({}^{i}I_i{}^{i}\omega_i) \\
{}^{i}f_i &= {}^{i}F_i + A_{i+1}({}^{i+1}f_{i+1})
\end{aligned}
$$

where $A_i^T = (A_i)^{-1}$, and ${}^{j}v$ denotes a vector v represented in the orientation of coordinate system j measured from the base origin. From Tables I and II it is seen that the recursive Newton-Euler formulation gives a 60% reduction in additions and multiplications over the recursive Lagrangian formulation. For $n = 6$ there are only 852 multiplications and 738 additions.

The Configuration Space Method

The Configuration Space Method [6] is derived from the following reformulation of the Lagrangian dynamics:

$$F_i = G_i(q) + \sum_{j=1}^{n} J_{ij}(q)\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ijk}\dot{q}_j\dot{q}_k$$

The gravity compensation G_i, inertial terms J_{ij}, and Coriolis coefficients C_{ijk} are tabulated in this formulation. The number of additions and multiplications using the Configuration Space Method are presented in Tables I and II. Although there is still an n^3 dependence for the additions and multiplications, the much smaller coefficients on the polynomial terms represent a greatly reduced computational cost. For $n = 6$ there are just 468 multiplications and 264 additions. An additional saving not indicated in these figures is that it is unnecessary to compute the sines and cosines of angles. However, for $n \geq 9$ the Configuration Space Method becomes less efficient than the Newton-Euler formulation.

Conclusions

The problem of computing manipulator dynamics in real time appears to have been solved. With both the recursive Lagrangian formulation and the recursive Newtonian formulation of the dynamics, the number of additions and multiplications varies linearly with the number of joints. For 6 joints the number of additions and multiplications should be computable in real time with even modest computers such as PDP11's. The impetus for applying approximations or tabular techniques to solve the dynamics appears to have been lost. With respect to the Configuration Space Method in particular, which for $n < 9$ joints is the most efficient formulation, problems incumbent with the use of tables such as large memories, configuration sensitivity, interpolation, and filling the tables are obviated.

It appears unlikely that any but very minor improvements in efficiency can be made to the present recursive formulations. The recursive Newton-Euler formulation is more efficient than the recursive Lagrangian

formulation presented here. One reason for presenting this recursive Lagrangian formulation is to demonstrate that the Lagrangian formulation is not necessarily so computationally intensive as to preclude real time computation, as had been the assumption for the past 10 years [2,3,4,6], and to demonstrate that the Lagrangian formulation can be made roughly as efficient as the Newton-Euler formulation [5]. For some applications involving the use of homogenous coordinates and 4x4 transformation matrices, e.g. [23], the recursive Lagrangian formulation may be the most convenient efficient dynamics formulation.

References

[1] J. J. Uicker, "On the Dynamic Analysis of Spatial Linkages Using 4x4 Matrices," Ph. D. Thesis, Northwestern University, August 1965.

[2] M. E. Kahn, "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains," Stanford Artificial Intelligence Project Memo AIM-106, December 1969.

[3] R. Paul, "Modelling, Trajectory Calculation, and Servoing of a Computer Controlled Arm," A.I. Memo 177, Stanford Artificial Intelligence Laboratory, September 1972.

[4] A. K. Bejczy, "Robot Arm Dynamics and Control," JPL Technical Memorandum 33-669, February 1974.

[5] J. Luh, M. Walker and R. Paul, "On-line Computational Scheme for Mechanical Manipulators," 2nd IFAC/IFIP Symposium on Information Control Problems in Manufacturing Technology, Stuttgart, Germany, October 22-24, 1979.

[6] B. K. P. Horn and M. H. Raibert, "Configuration Space Control," **The Industrial Robot**, pp. 69-73, June, 1978.

[7] M. H. Raibert, "Analytical Equations Vs. Table Look-up for Manipulation: a Unifying Concept," **Proc. IEEE Conf. on Decision and Control**, New Orleans, pp. 576-579, December 1977.

[8] J. S. Albus, "A New Approach to Manipulator Control: the Cerebellar Model Articulation Controller (CMAC)," **J. Dynamic Systems, Measurement, and Control**, vol. 97, pp. 270-277, 1975.

[9] J. S. Albus, "Data Storage in the Cerebellar Model Articulation Controller (CMAC)," **J. Dynamic Systems, Measurement, and Control**, vol. 97, pp. 228-233, 1975.

[10] M. H. Raibert, "A Model for Sensorimotor Control and Learning," **Biol. Cyber.**, vol. 29, pp. 29-36, 1978.

[11] W. W. Armstrong, "Recursive Solution to the Equations of Motion of an N-Link Manipulator," **Proc. 5th World Congress on Theory of Machines and Mechanisms**, vol. 2, pp. 1343-1346, July 1979.

[12] D. E. Orin, R. B. McGhee, M. Vukobratovic, and G. Hartoch, "Kinematic and Kinetic Analysis of Open-Chain Linkages Utilizing Newton-Euler Methods," **Math. Biosc.**, vol. 43, pp. 107-130, 1979.

[13] R. C. Waters, "Mechanical Arm Control," MIT Artificial Intelligence Laboratory Memo 549, October 1979.

[14] W. W. Hooker and G. Margulies, "The Dynamical Attitude Equations for an n-Body Satellite," **J. Astronautical Sciences**, vol. 12 no. 4, pp. 123-128, 1965.

[15] W. W. Hooker, "A Set of r Dynamical Attitude Equations for an Arbitrary n-Body Satellite Having r Rotational Degrees of Freedom," **AIAA Journal**, vol. 8 no. 7, pp. 1205-1207, 1970.

[16] Y. Stepanenko, and M. Vukobratovic, "Dynamics of Articulated Open-Chain Active Mechanisms," **Math. Biosc.**, vol. 28, pp. 137-170, 1976.

[17] M. Vukobratovic, "Dynamics of Active Articulated Mechanisms and Synthesis of Artificial Motion," **Mechanism and Machine Theory**, vol 13, pp. 1-56, 1978.

[18] M. Vukobratovic, D. Stokic, and D. Hristic, "New Control Concept of Anthropomorphic Manipulators," **Mechanism and Machine Theory**, vol. 12, pp. 515-530, 1977.

[19] T. R. Kane, "Dynamics of Nonholonomic Systems," **J. Applied Mechanics**, vol. 28, pp. 574-578, 1961.

[20] R. L. Huston, C. E. Passerello, and M. W. Harlow, "Dynamics of Multirigid-Body Systems," **J. Applied Mechanics**, vol. 45, pp. 889-894, 1978.

[21] Wittenburg, J., **Dynamics of Systems of Rigid Bodies**. Stuttgart: B.G. Teubner, 1977.

[22] H. Hemami, V. C. Jaswa, and R. B. McGhee, "Some Alternative Formulations of Manipulator Dynamics for Computer Simulation Studies," Proc. 13th Allerton Conference on Circuit and System Theory, U. Illinois, pp.124-140, October 1975.

[23] R. Paul, "Manipulator Cartesian Path Control," **IEEE Trans. Systems, Man, Cyb.**, vol. 9 no. 2, pp. 702-711, 1979.

Appendix A

In this appendix the kinetic energy for a link is derived, and the Lagrange equations are solved. Let h_i be a vector from the base origin to a point fixed in link i, h_i^* a vector from coordinate system i to the same point. Then

$$h_i = p_i + W_i\, {}^i h_i^* \tag{A1}$$

$$\dot{h}_i = \dot{p}_i + \dot{W}_i\, {}^i h_i^* \tag{A2}$$

The kinetic energy for a particle on link i is given by:

$$\begin{aligned} k_i &= \frac{1}{2} tr\left(\dot{h}_i \dot{h}_i^T\right) dm \\ &= \frac{1}{2} tr\left(\dot{p}_i \dot{p}_i^T + 2\dot{W}_i\, {}^i h_i^* \dot{p}_i^T + \dot{W}_i\, {}^i h_i^*\, {}^i h_i^{*T} \dot{W}_i^T\right) dm \end{aligned} \tag{A3}$$

Integrating over all particles in link i, the total kinetic energy K_i of link i is given by:

$$K_i = \frac{1}{2} tr\left(m_i \dot{p}_i \dot{p}_i^T + 2\dot{W}_i\, {}^i n_i \dot{p}_i^T + \dot{W}_i J_i \dot{W}_i^T\right) \tag{A4}$$

where m_i is the mass of link i, ${}^i n_i = \int {}^i h_i^* dm$ and ${}^i n_i / m_i$ is the vector to the center of mass, and $J_i = \int {}^i h_i^*\, {}^i h_i^{*T} dm$ is the inertia tensor with respect to the coordinate system i. The total kinetic energy for all the links is given by:

$$\begin{aligned} K &= \sum_{j=1}^{n} K_j \\ &= \frac{1}{2} \sum_{j=1}^{n} tr\left(m_j \dot{p}_j \dot{p}_j^T + 2\dot{W}_j\, {}^j n_j \dot{p}_j^T + \dot{W}_j J_j \dot{W}_j^T\right) \end{aligned} \tag{A5}$$

The potential energy derives from gravity and is equal to the sum of the work required to displace the mass center of each link from the horizontal reference plane:

$$P = \bar{P} - \sum_{j=1}^{n} m_i g^T W_i \, {}^i r_i \tag{A6}$$

where $g = (g_x \, g_y \, g_z)$ is the acceleration due to gravity and $\bar{P}$ is a constant which depends on the particular reference plane chosen. The Lagrangian L is defined as the difference between the kinetic and potential energies: $L = K - P$. The Lagrange equations are:

$$F_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \qquad i = 1, \ldots, n \tag{A7}$$

Since the potential energy is position dependent only,

$$F_i = \frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i} \tag{A8}$$

The terms are calculated below.

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) = \sum_{j=i}^{n} tr\left(m_j \frac{\partial \dot{p}_j}{\partial \dot{q}_i} \ddot{p}_j^T + m_j \frac{d}{dt}\left(\frac{\partial \dot{p}_j}{\partial \dot{q}_i}\right) \dot{p}_j^T + \frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right) {}^j n_j \dot{p}_j^T + \frac{\partial \dot{W}_j}{\partial \dot{q}_i} \, {}^j n_j \ddot{p}_j^T \right.$$

$$\left. + \ddot{W}_j \, {}^j n_j \frac{\partial \dot{p}_j^T}{\partial \dot{q}_i} + \dot{W}_j \, {}^j n_j \frac{d}{dt}\left(\frac{\partial \dot{p}_j^T}{\partial \dot{q}_i}\right) + \frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \ddot{W}_j^T + \frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right) J_j \dot{W}_j^T \right) \tag{A9}$$

$$\frac{\partial K}{\partial q_i} = \sum_{j=i}^{n} tr\left(m_j \frac{\partial \dot{p}_j}{\partial q_i} \dot{p}_j^T + \frac{\partial \dot{W}_j}{\partial q_i} \, {}^j n_j \dot{p}_j^T + \dot{W}_j \, {}^j n_j \frac{\partial \dot{p}_j^T}{\partial \dot{q}_i} + \frac{\partial \dot{W}_j}{\partial q_i} J_j \dot{W}_j^T \right) \tag{A10}$$

$$\frac{\partial P}{\partial q_i} = -\sum_{j=i}^{n} m_j g^T \frac{\partial W_j}{\partial q_i} \, {}^j r_j \tag{A11}$$

It can be shown that

$$\frac{\partial p_j}{\partial q_i} = \frac{\partial \dot{p}_j}{\partial \dot{q}_i}$$

$$\frac{\partial \dot{p}_j}{\partial q_i} = \frac{d}{dt}\left(\frac{\partial \dot{p}_j}{\partial \dot{q}_i}\right)$$

$$\frac{\partial W_j}{\partial q_i} = \frac{\partial \dot{W}_j}{\partial \dot{q}_i} \qquad (A12)$$

$$\frac{\partial \dot{W}_j}{\partial q_i} = \frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right)$$

Substituting into the Lagrangian, and noting that $tr(AB^T) = tr(BA^T)$,

$$F_i = \sum_{j=i}^{n}\left[tr\left(m_j \frac{\partial p_j}{\partial q_i}\ddot{p}_j^T + \frac{\partial p_j}{\partial q_i}\,{}^j n_j^T \ddot{W}_j^T + \frac{\partial W_j}{\partial q_i}\,{}^j n_j \ddot{p}_j^T + \frac{\partial W_j}{\partial q_i} J_j \ddot{W}_j^T\right) - m_j g^T \frac{\partial W_j}{\partial q_i}\,{}^j r_j\right] \qquad (A13)$$

Appendix B

In this appendix the forward recursive relations are developed. For $i \leq j$,

$$
\begin{aligned}
{}^{i}p_j &= {}^{i}p_i + {}^{i}W_i\, {}^{i}p_j \\
\frac{\partial {}^{i}p_j}{\partial q_i} &= \frac{\partial {}^{i}W_i}{\partial q_i}\, {}^{i}p_j \\
\frac{\partial {}^{i}W_j}{\partial q_i} &= \frac{\partial {}^{i}W_i}{\partial q_i}\, {}^{i}W_j
\end{aligned}
\tag{B1}
$$

Substituting into (15) and factoring,

$$
F_i = tr\left(\frac{\partial {}^{i}W_i}{\partial q_i} \sum_{j=i}^{n} \left(m_j\, {}^{i}p_j \ddot{p}_j^T + {}^{i}p_j\, {}^{j}n_j^T \ddot{W}_j^T + {}^{i}W_j\, {}^{j}n_j \ddot{p}_j^T + {}^{i}W_j J_j \ddot{W}_j^T \right)\right) - g^T \frac{\partial {}^{i}W_i}{\partial q_i} \sum_{j=i}^{n} m_j\, {}^{i}W_j\, {}^{j}r_j
\tag{B2}
$$

The terms inside the summations may be computed by recurrence relations.

$$
\begin{aligned}
D_i &= \sum_{j=i}^{n} \left(m_j\, {}^{i}p_j \ddot{p}_j^T + {}^{i}p_j\, {}^{j}n_j^T \ddot{W}_j^T + {}^{i}W_j\, {}^{j}n_j \ddot{p}_j^T + {}^{i}W_j J_j \ddot{W}_j^T \right) \\
&= \sum_{j=i+1}^{n} \left(\left(A_{i+1}\, {}^{i+1}p_j + {}^{i}p_{i+1} \right) \left(m_j \ddot{p}_j^T + {}^{j}n_j^T \ddot{W}_j^T \right) + A_{i+1}\, {}^{i+1}W_j \left({}^{j}n_j \ddot{p}_j^T + J_j \ddot{W}_j^T \right) \right) + {}^{i}n_i \ddot{p}_i^T + J_i \ddot{W}_i^T \\
&= A_{i+1} D_{i+1} + {}^{i}p_{i+1} e_{i+1} + {}^{i}n_i \ddot{p}_i^T + J_i \ddot{W}_i^T
\end{aligned}
\tag{B3}
$$

where

$$
\begin{aligned}
e_i &= \sum_{j=i}^{n} \left(m_j \ddot{p}_j^T + {}^{j}n_j^T \ddot{W}_j^T \right) \\
&= e_{i+1} + m_i \ddot{p}_i^T + {}^{i}n_i^T \ddot{W}_i^T
\end{aligned}
\tag{B4}
$$

The recurrence relation c_i for the gravity term is the same as equation (13), except for the dimensionality difference. Substituting into (B2), the generalized force is:

$$F_i = tr\left(\frac{\partial W_i}{\partial q_i} D_i\right) - g^T \frac{\partial W_i}{\partial q_i} c_i \qquad (B5)$$